CAPÍTULO 09

codea
Cursos de programación con proyectos

Gestión de Servicios

Implementar el CRUD completo del modelo Service, configurando su relación uno a muchos con el modelo Category, y habilitando funcionalidades SEO e imagen.

Puntos esenciales

- **Relación**: Cada *servicio* pertenece a una *categoría* (`belongsTo`), y una *categoría* puede tener muchos *servicios* (`hasMany`).
- **Migraciones y modelos**: Se crea las tabla `services` con clave foránea y campos SEO. Se configuran `Service` y `Category` con relaciones Eloquent.
- **Controlador CRUD**: `ServiceController` permite crear, listar, editar, ver y eliminar servicios.
- **Rutas RESTful**: Declaradas con `Route::resource`.
- **Vistas Blade**: Formulario de creación, edición, detalle y listado con Bootstrap.
- **Imágenes y SEO**: Soporte para imágenes públicas y campos `meta_title` y `meta_description`.
- **Integración con Categorías**: Al seleccionar una categoría, se guardará su ID en sesión para mostrar sus servicios.

📁 Estructura del módulo

```
app/
├── Models/
│   ├── Category.php            ← Modelo de categoría
│   └── Service.php             ← Modelo de servicio con relación belongsTo

app/Http/Controllers/
└── Admin/
    └── ServiceController.php   ← Controlador CRUD para servicios

resources/
└── views/
    └── admin/
        ├── services/
        │   ├── index.blade.php     ← Lista de servicios
        │   ├── create.blade.php    ← Formulario de creación
        │   └── edit.blade.php      ← Formulario de edición
        └── partials/
            └── message.blade.php   ← Mensajes flash y errores
```

📁 Estructura del módulo

```
routes/
└── web.php                          ← Definición de rutas admin.services

database/
└── migrations/
    ├── xxxx_xx_xx_create_categories_table.php
    └── xxxx_xx_xx_create_services_table.php

public/
└── storage/                         ← Carpeta pública para imágenes de servicios (via storage:link)

storage/
└── app/
    └── public/
        └── services/                ← Carpeta donde se guardan las imágenes
```

Modelo y la migración Servicios

Modelo: **Service|** Tabla: **Services**

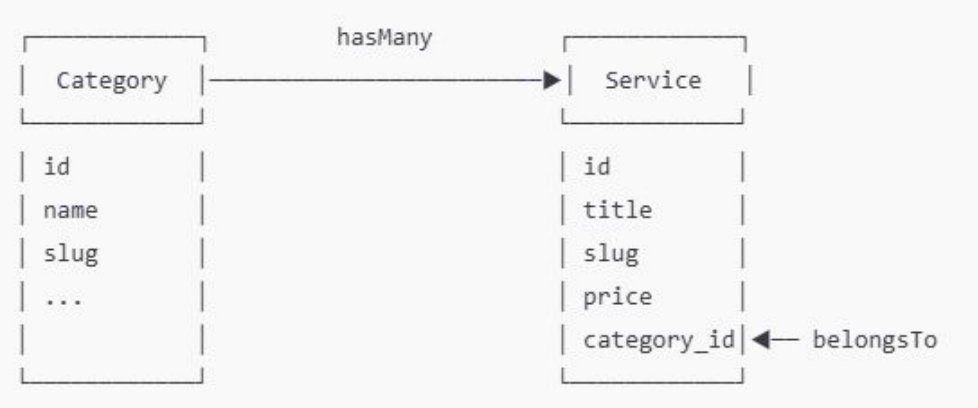
Campos:

- `id`: Identificador único (auto-incremental).
- `title`: Nombre del servicio.
- `slug`: URL amigable, única.
- `description`: Detalle del servicio.
- `price`: Precio con 2 decimales.
- `image_url`: URL de la imagen, opcional.
- `category_id`: Clave foránea que referencia a `categories`.
- `meta_title` y `meta_description`: Campos SEO, opcionales.
- `timestamps`: Fechas de creación y actualización.

📌 *Importante:* El slug debe ser único y generado automáticamente.

- php artisan make:model Service -m
- php artisan migrate

Establecer la relación entre los Modelos



🧠 En código Laravel

En `Category.php`:

```
public function services(){
 return $this->hasMany(Service::class);
}
```

En `Service.php`:

```
public function category(){
   return $this->belongsTo(Category::class);
}
```

Controlador CategoryController

Modificar el `CategoryController` para Almacenar el `category_id` en la Sesión

Acciones:

- Vista index
- Método Show

Controlador ServiceController

Acciones:

- index()
- create()
- store()
- edit()
- update()
- destroy()

Vistas Blade

Todas las vistas del CRUD de servicios **extienden un layout principal** ubicado en:

`resources/views/layouts/admin.blade.php`

- **index**: Tabla con botones
- **create**: Formulario nuevo
- **edit**: Formulario de edición

📌 *Vistas organizadas en `admin/service/`.*

✅ Resultado Final

Con esta lección, ahora puedes:

- Crear y modificar categorías desde el panel Admin.
- Generar slugs SEO automáticamente.
- Usar rutas protegidas por middleware.
- Administrar el contenido dinámicamente.

codea
Cursos de programación con proyectos

FIN

Curso completo : Laravel Página Web Administrable